Boulder

Recommender Systems

YouTube Playlist

Maziar Raissi

Assistant Professor

Department of Applied Mathematics

University of Colorado Boulder

maziar.raissi@colorado.edu

Session-based Recommendations with Recurrent Neural Networks

Boulder

YouTube Playlist

– instead of long user histories (as in the case of Netflix)
– short session-based data (e.g. a small sportsware website)
– Factor models (sparse user-item interaction matrix)
– Neighborhood methods (co-occurrences of items in sessions)

Gated Recurrent Unit (GRU)

$h_t = (1 - z_t)h_{t-1} + z_t\tilde{h}_t$

$z_t = \sigma(W_z x_t + U_z h_{t-1}) \rightarrow$ update gate

$\tilde{h}_t = \tanh(W x_t + U(r_t \odot h_{t-1}))$

$r_t = \sigma(W_r x_t + U_r h_{t-1}) \rightarrow$ reset gate

Input: actual state of the session

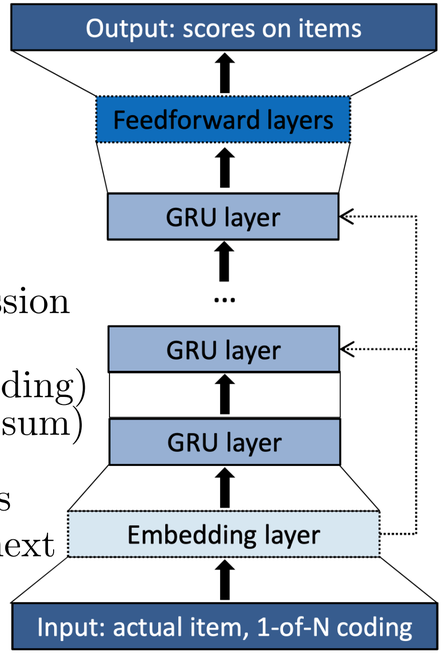
Output: item of the next event in the session

State of the session:

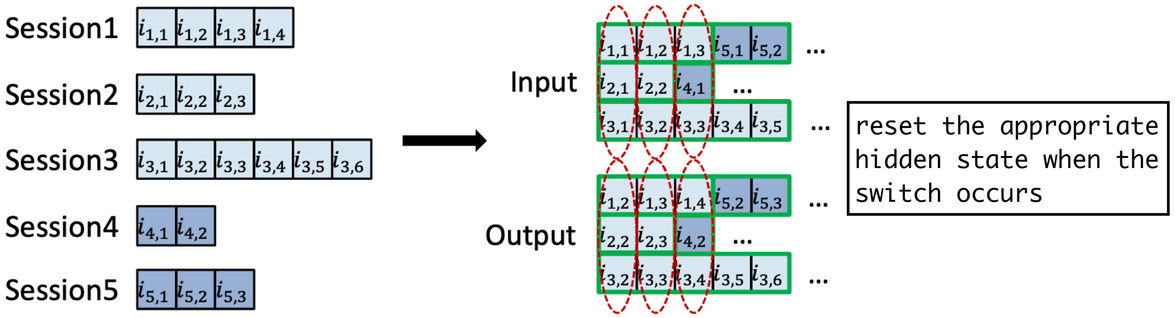
- item of the actual event (1-of-N encoding)
- events in the session so far (weighted sum)

$N \rightarrow$ number of items

Output: predicted preference of the items (i.e. the likelihood of being the next in the session for each item)



Session-parallel mini-batches



Ranking Loss

BPR: Bayesian Personalized Ranking

user prefers item i over item j

$i \rightarrow$ desired item (next item in the session)

$j \rightarrow$ negative samples

$\hat{r}_s \rightarrow$ scores at a given point of the session s

$$L_s = -\frac{1}{N_S}\sum_{j=1}^{N_S}\log(\sigma(\hat{r}_{s,i} - \hat{r}_{s,j}))$$

$L_s \rightarrow$ loss at a given point of the session s

$N_S \rightarrow$ sample size

TOP1 loss

$$\frac{1}{N}\sum_{j=1}^{N} I\{\hat{r}_{s,j} > \hat{r}_{s,i}\}$$

relative rank of the relevant item

– regularized approximation

$$L_s = \frac{1}{N_S} \cdot \sum_{j=1}^{N_S} \sigma\left(\hat{r}_{s,j} - \hat{r}_{s,i}\right) + \sigma\left(\hat{r}_{s,j}^2\right)$$

Evaluation

Recall@20: proportion of cases having the desired item amongst the top-20 items in all test cases

MRR@20: Mean Reciprocal Rank (average of the reciprocal ranks of the desired items)

Loss / #Units	RSC15 Recall@20	RSC15 MRR@20
TOP1 100	0.5853 (+15.55%)	0.2305 (+12.58%)
BPR 100	0.6069 (+19.82%)	0.2407 (+17.54%)
Cross-entropy 100	0.6074 (+19.91%)	0.2430 (+18.65%)
TOP1 1000	0.6206 (+22.53%)	**0.2693 (+31.49%)**
BPR 1000	**0.6322 (+24.82%)**	0.2467 (+20.47%)
Cross-entropy 1000	0.5777 (+14.06%)	0.2153 (+5.16%)

Compared to the best baseline (item-KNN)

VIDEO Recall@20	VIDEO MRR@20
0.6141 (+11.50%)	0.3511 (+3.84%)
0.5999 (+8.92%)	0.3260 (-3.56%)
0.6372 (+15.69%)	0.3720 (+10.04%)
0.6624 (+20.27%)	**0.3891 (+15.08%)**
0.6311 (+14.58%)	0.3136 (-7.23%)
–	–

Baseline	RSC15 Recall@20	RSC15 MRR@20	VIDEO Recall@20	VIDEO MRR@20
POP	0.0050	0.0012	0.0499	0.0117
S-POP	0.2672	0.1775	0.1301	0.0863
Item-KNN	0.5065	0.2048	0.5508	0.3381
BPR-MF	0.2574	0.0618	0.0692	0.0374

Best parametrizations for datasets/loss functions

Dataset	Loss	Mini-batch	Dropout	Learning rate	Momentum
RSC15	TOP1	50	0.5	0.01	0
RSC15	BPR	50	0.2	0.05	0.2
RSC15	Cross-entropy	500	0	0.01	0
VIDEO	TOP1	50	0.4	0.05	0
VIDEO	BPR	50	0.3	0.1	0
VIDEO	Cross-entropy	200	0.1	0.05	0.3

VIDEO: YouTube-like OTT video service platform

RSC15: RecSys Chalenge 2015

Hidasi, Balázs, et al. "Session-based recommendations with recurrent neural networks." *arXiv preprint arXiv:1511.06939* (2015).

AutoRec: Autoencoders Meet Collaborative Filtering

YouTube Video

Collaborative filtering (CF): exploit information about users' preferences for items (e.g. star ratings) to provide personalised recommendations.
– matrix factorisation
– neighbourhood models

– rating-based collaborative filtering
$m \rightarrow$ number of users
$n \rightarrow$ number of items
$R \in \mathbb{R}^{m \times n} \rightarrow$ user-item rating matrix
$u \in U = \{1, 2, \ldots, m\} \rightarrow$ user
$i \in I = \{1, 2, \ldots, n\} \rightarrow$ item
$r^u = (R_{u1}, \ldots, R_{un}) \in \mathbb{R}^n$
partially observed vector (representing user u)
$r^i = (R_{1i}, \ldots, R_{mi}) \in \mathbb{R}^m$
partially observed vector (representing item i)

Item-based AutoRec

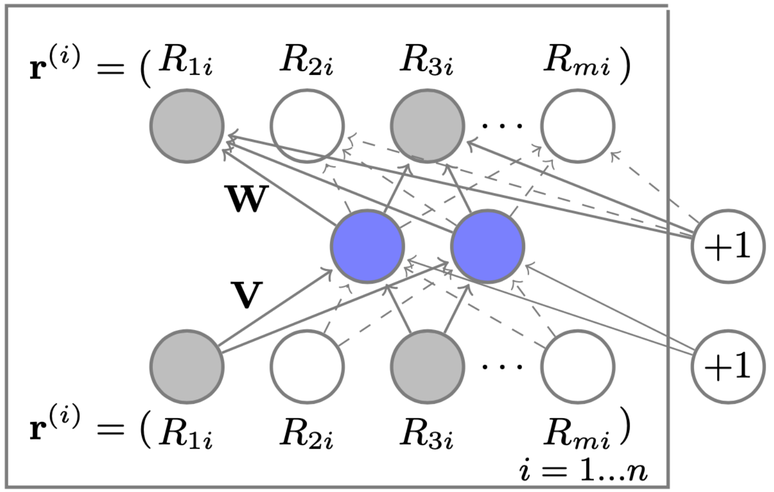
$$\min_\theta \sum_{i=1}^{n} \|r^i - h(r^i; \theta)\|_{\mathcal{O}}^2 + \frac{\lambda}{2}(\|W\|_F^2 + \|V\|_F^2)$$

$$h(r; \theta) = f(W g(V r + \mu) + b)$$

$\|\cdot\|_{\mathcal{O}} \rightarrow$ consider only the contribution of observed ratings

$$\hat{R}_{ui} = h(r^i; \theta)_u$$

User-based AutoRec $\rightarrow$ similar

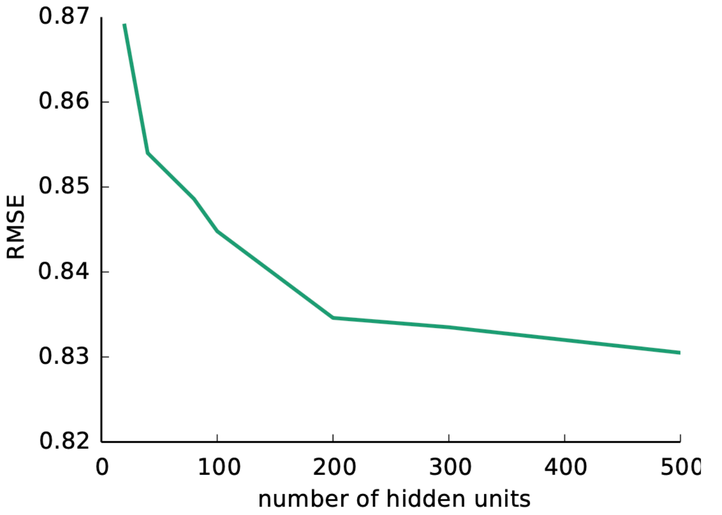


	ML-1M	ML-10M	Netflix
BiasedMF	0.845	0.803	0.844
I-RBM	0.854	0.825	-
U-RBM	0.881	0.823	0.845
LLORMA	0.833	**0.782**	0.834
I-AutoRec	**0.831**	**0.782**	**0.823**

Movielens 1M & 10M

RMSE	ML-1M	ML-10M
U-AutoRec	0.874	0.867
I-AutoRec	**0.831**	**0.782**

$f(\cdot)$	$g(\cdot)$	RMSE
Identity	Identity	0.872
Sigmoid	Identity	0.852
Identity	Sigmoid	**0.831**
Sigmoid	Sigmoid	0.836



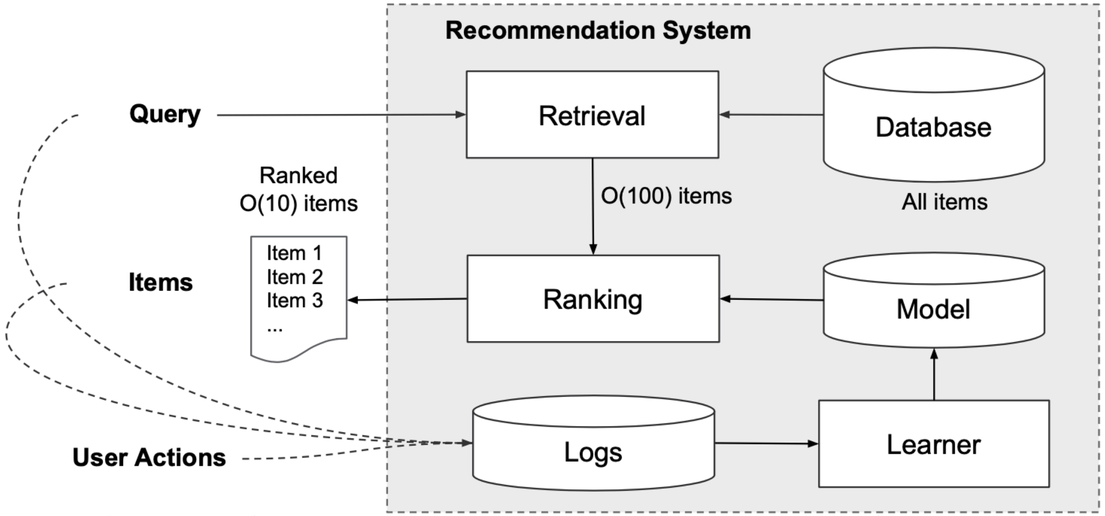
Sedhain, Suvash, et al. "Autorec: Autoencoders meet collaborative filtering." *Proceedings of the 24th international conference on World Wide Web*. 2015.

Wide & Deep Learning for Recommender Systems

YouTube Video

Google Play: one billon active users & one million apps
Impressions: list of apps

"A recommender system can be viewed as a search ranking system, where the input query is a set of user and contextual information, and the output is a ranked list of items"



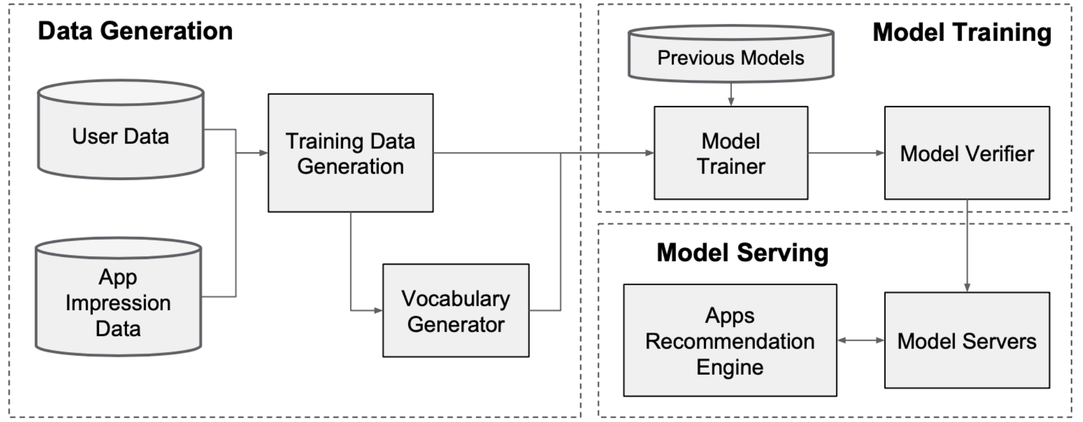
– user features (e.g., country, language, demographics)
– contextual features (e.g., device, hour of the day, day of the week)
– impression features (e.g., app age, historical statistics of an app)

The label is app acquisition: 1 if the impressed app was installed, and 0 otherwise

$p(y = 1|x) = \sigma(w_{\text{wide}}^T[x, \phi(x)] + w_{\text{deep}}^T a^L + b)$

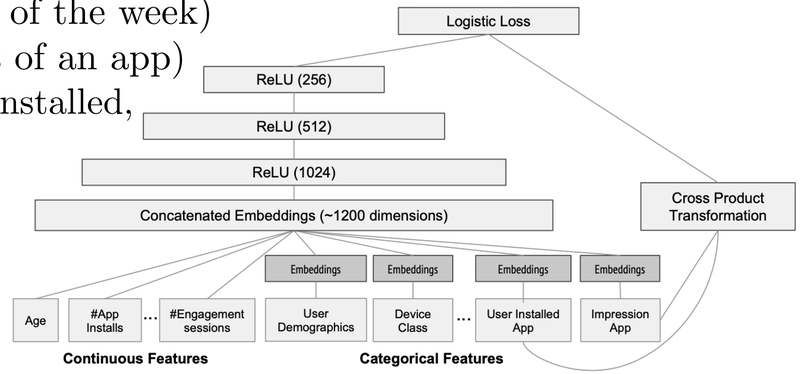
$x = [x_1, x_2, ..., x_d]$ is a vector of d features

$$\phi_k(x) = \prod_{i=1}^{d} x_i^{c_{ki}}, c_{ki} \in \{0, 1\} \qquad a^{l+1} = f(W^l a^l + b^l)$$



For binary features, a cross-product transformation (e.g., "AND(gender=female, language=en)") is 1 if and only if the constituent features ("gender=female" and "language=en") are all 1, and 0 otherwise.

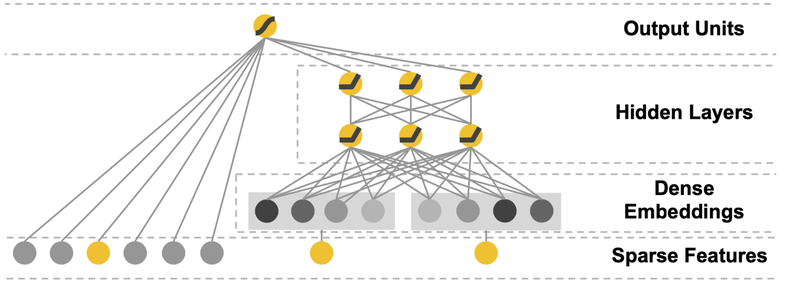
Continuous real-valued features are normalized to $[0, 1]$ by mapping a feature value x to its cumulative distribution function $P(X \leq x)$, divided into n_q quantiles. The normalized value is $(i - 1)/(1 - n_q)$ for values in the i-th quantiles.



– memorization
– generalization

Model	Offline AUC
Wide (control)	0.726
Deep	0.722
Wide & Deep	0.728

Online Acquisition Gain
0%
+2.9%
+3.9%



Cheng, Heng-Tze, et al. "Wide & deep learning for recommender systems." *Proceedings of the 1st workshop on deep learning for recommender systems*. 2016.

Neural Collaborative Filtering

YouTube Video

Collaborative Filtering: modeling users' preference on items based on their past interactions (e.g., ratings and clicks)

Learning from implicit data

$M \rightarrow$ number of users

$N \rightarrow$ number of items

$Y \in \mathbb{R}^{M \times N} \rightarrow$ user-item interaction matrix

$y_{ui} = 1$ if interaction (user u, item i) is observed

doesn't mean that user u likes item i

$y_{ui} = 0$ otherwise

doesn't mean that user u didn't like item i

$\hat{y}_{ui} = f(u, i; \theta)$

$\hat{y}_{ui}$: predicted score of interaction y_{ui}; f: interaction function

pointwise loss $\rightarrow$ squared loss btw $\hat{y}_{ui}$ & y_{ui}

pairwise loss $\rightarrow$ maximizes the margin btw $\hat{y}_{ui}$ & y_{uj}

Matrix Factorization (MF)

$p_u \rightarrow$ latent vector for user u

$q_i \rightarrow$ latent vector for item i

$$\hat{y}_{ui} = f(u, i; p_u, q_i) = p_u^T q_i = \sum_{k=1}^{K} p_{uk} q_{ik}$$

$K \rightarrow$ dimension of the latent space

Neural Collaborative Filtering

$v_u^U \rightarrow$ one-hot encoding feature vector for user u

$v_i^I \rightarrow$ one-hot encoding feature vector for item i

$$\hat{y}_{ui} = f(P^T v_u^U, Q^T v_i^I; P, Q, \theta_f)$$

$$L_{sqr} = \sum_{(u,i) \in \mathcal{Y} \cup \mathcal{Y}^-} w_{ui} (y_{ui} - \hat{y}_{ui})^2$$

$\mathcal{Y} \rightarrow$ set of observed interactions in Y

$\mathcal{Y}^- \rightarrow$ set of negative instances

all (or sampled from) unobserved interactions

$w_{ui} \rightarrow$ weight of training instance

Likelihood

$$p(\mathcal{Y}, \mathcal{Y}^-; P, Q, \theta_f) = \prod_{(u,i) \in \mathcal{Y}} \hat{y}_{ui} \prod_{(u,j) \in \mathcal{Y}^-} (1 - \hat{y}_{uj})$$

$\hat{y}_{ui} \rightarrow$ how likely i is relevant to u

$$L = -\sum_{(u,i) \in \mathcal{Y}} \log \hat{y}_{ui} - \sum_{(u,j) \in \mathcal{Y}^-} \log(1 - \hat{y}_{uj})$$

$$= -\sum_{(u,i) \in \mathcal{Y} \cup \mathcal{Y}^-} y_{ui} \log \hat{y}_{ui} + (1 - y_{ui}) \log(1 - \hat{y}_{ui})$$

Generalized Matrix Factorization (GMF)

$$p_u = P^T v_u^U$$

$$q_i = Q^T v_i^I$$

$$\hat{y}_{ui} = a_{\text{out}}(h^T (p_u \odot q_i))$$

If $h = 1$ and $a_{\text{out}} \equiv \text{id} \implies \text{GMF} \equiv \text{MF}$

Fusion of GMF and MLP (NeuMF)

$$\phi^{GMF} = \mathbf{p}_u^G \odot \mathbf{q}_i^G,$$

$$\phi^{MLP} = a_L(\mathbf{W}_L^T(a_{L-1}(...a_2(\mathbf{W}_2^T \begin{bmatrix} \mathbf{p}_u^M \\ \mathbf{q}_i^M \end{bmatrix} + \mathbf{b}_2)...)) + \mathbf{b}_L)$$

$$\hat{y}_{ui} = \sigma(\mathbf{h}^T \begin{bmatrix} \phi^{GMF} \\ \phi^{MLP} \end{bmatrix}),$$

Multi-Layer Perceptron (MLP)

$$\mathbf{z}_1 = \phi_1(\mathbf{p}_u, \mathbf{q}_i) = \begin{bmatrix} \mathbf{p}_u \\ \mathbf{q}_i \end{bmatrix},$$

$$\phi_2(\mathbf{z}_1) = a_2(\mathbf{W}_2^T \mathbf{z}_1 + \mathbf{b}_2),$$

.....

$$\phi_L(\mathbf{z}_{L-1}) = a_L(\mathbf{W}_L^T \mathbf{z}_{L-1} + \mathbf{b}_L),$$

$$\hat{y}_{ui} = \sigma(\mathbf{h}^T \phi_L(\mathbf{z}_{L-1})),$$

Evaluation

$$\text{Hit@}\ell = \frac{1}{m} \sum_{u \in \mathcal{U}} \mathbb{1}\{\text{rank}_{u,g_u} \leq \ell\}$$

$\text{rank}_{u,g_u} \rightarrow$ ranking of the ground-truth item g_u

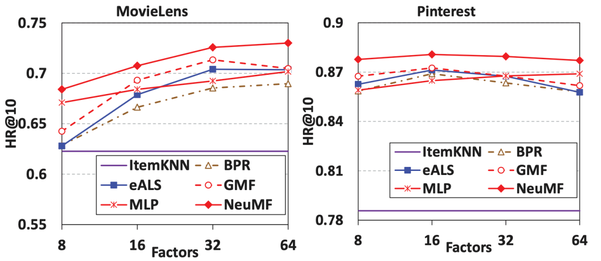
$\mathcal{U} \rightarrow$ user set

$m \rightarrow$ number of users

$$\text{AUC} = \frac{1}{m} \sum_{u \in \mathbb{U}} \frac{1}{|I \setminus S_u|} \sum_{j \in I \setminus S_u} \mathbb{1}\{\text{rank}_{u,g_u} < \text{rank}_{u,j}\}$$

$I \rightarrow$ item set

$S_u \rightarrow$ candidate items of user u



He, Xiangnan, et al. "Neural collaborative filtering." Proceedings of the 26th international conference on world wide web. 2017.

Neural Factorization Machines for Sparse Predictive Analytics

YouTube Video

– recommendation systems – targeted advertising
– search ranking – visual analysis – event detection

Online advertising: predict $\overbrace{\text{how likely}}^{\text{target}}$ $\overbrace{\text{a user}}^{\text{1st predictor}}$ of a particular $\underbrace{\text{occupation}}_{\text{2nd predictor}}$ will click on $\underbrace{\text{an ad}}_{\text{3rd predictor}}$

user IDs & demographics (e.g., genders & occupations)
one-hot encoding

Interaction btw features (cross features)

occupation $= \{\text{banker}, \text{doctor}\}$
gender $= \{\text{M}, \text{F}\}$
occupation_gender $= \{\text{banker\_M}, \text{banker\_F}, \text{doctor\_M}, \text{doctor\_F}\}$

Factorization Machines (FMs)

$x \in \mathbb{R}^n \rightarrow$ sparse real valued feature vector

$$\hat{y}_{\text{FM}} = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} v_i^T v_j x_i x_j$$

$v_i \in \mathbb{R}^k \rightarrow$ embedding vector for feature i
$k \rightarrow$ number of latent factors
Matrix Factorization: Models the relation of two entities only!

Neural Factorization Machines (NFMs)

$x_i = 0$ means i-th feature does not exist in the instance

$$\hat{y}_{\text{NFM}}(x) = w_0 + \sum_{i=1}^{n} w_i x_i + f(x)$$

Embedding Layer

$v_i \in \mathbb{R}^k \rightarrow$ embedding vector for the i-th feature
$\mathcal{V}_x = \{x_1 v_1, \ldots, x_n v_n\} \rightarrow$ set of embedding vectors (representing x)

Bi-Interaction Layer

$$f_{\text{BI}}(\mathcal{V}_x) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} x_i v_i \odot x_j v_j$$

Hidden Layers

$$\mathbf{z}_1 = \sigma_1(\mathbf{W}_1 f_{BI}(\mathcal{V}_x) + \mathbf{b}_1),$$
$$\mathbf{z}_2 = \sigma_2(\mathbf{W}_2 \mathbf{z}_1 + \mathbf{b}_2),$$
$$\ldots\ldots$$
$$\mathbf{z}_L = \sigma_L(\mathbf{W}_L \mathbf{z}_{L-1} + \mathbf{b}_L),$$

Prediction Layer

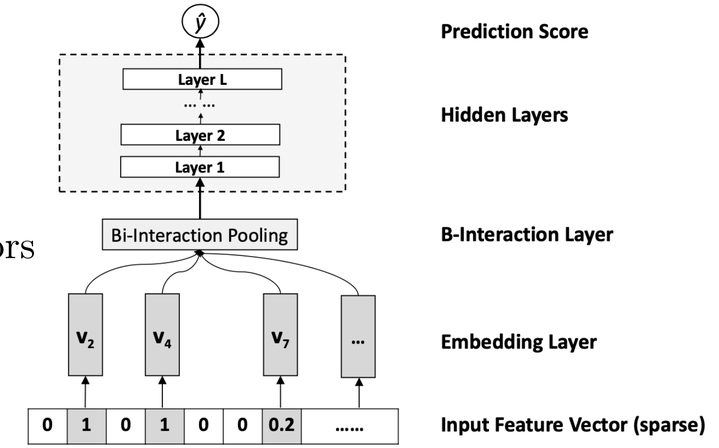
$$f(x) = h^T z_L$$

NFM generalizes FM

$$\hat{y}_{NFM-0} = w_0 + \sum_{i=1}^{n} w_i x_i + \mathbf{h}^T \sum_{i=1}^{n} \sum_{j=i+1}^{n} x_i \mathbf{v}_i \odot x_j \mathbf{v}_j$$

$$= w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \sum_{f=1}^{k} h_f v_{if} v_{jf} \cdot x_i x_j$$

$h = (1, 1, \ldots, 1) \implies$ FM

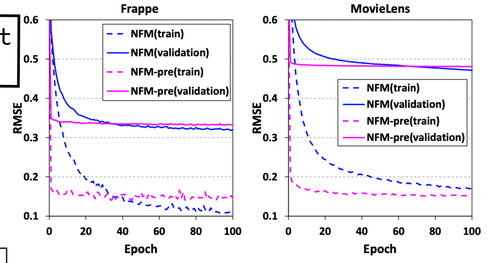


Learning

$$L_{\text{reg}} = \sum_{x \in \mathcal{X}} (\hat{y}(x) - y(x))^2$$

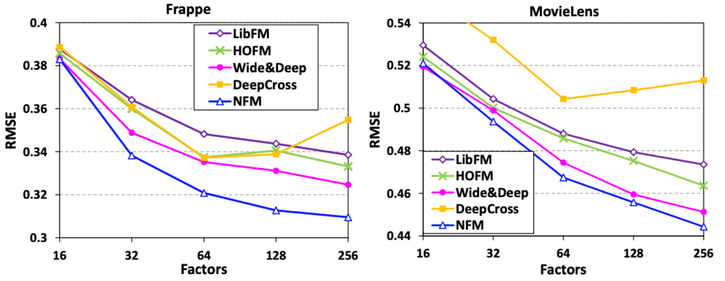
$\mathcal{X} \rightarrow$ set of instances for training

with and without pre-training



Statistics of the evaluation datasets.

Dataset	Instance#	Feature#	User#	Item#	
Frappe	288,609	5,382	957	4,082	→ context-aware app prediction
MovieLens	2,006,859	90,445	17,045	23,743	→ personalized tag recommendation

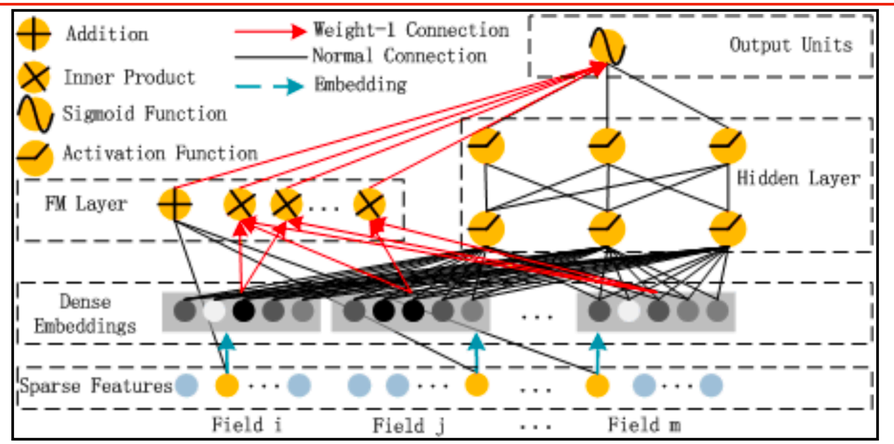


He, Xiangnan, and Tat-Seng Chua. "Neural factorization machines for sparse predictive analytics." Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval. 2017.

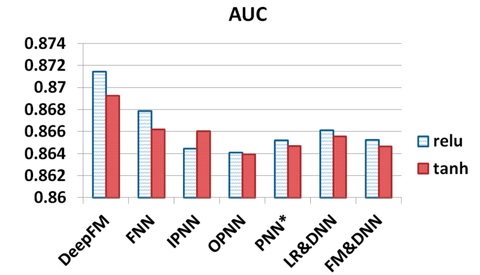
DeepFM: A Factorization-Machine based Neural Network for CTR Prediction

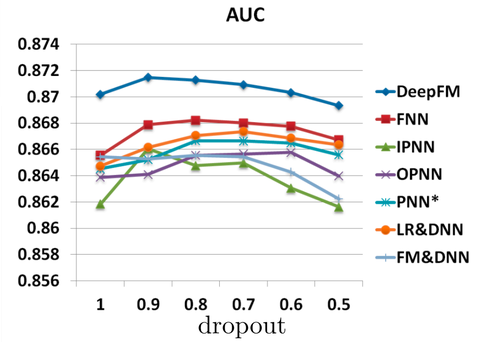
YouTube Video

CTR: Click Through Rate
Estimate the probability a user will click on a recommended item
CTR × bid → online advertising
bid → benefit the system receives if the item is clicked by a user
$n \rightarrow$ number of instances
$(\mathcal{X}, y) \rightarrow$ an instance
$\mathcal{X} \rightarrow m$-fields data record (usually involving a pair of user and item)
$y \in \{0, 1\} \rightarrow$ associated label indicating user behaviors
$y = 1 \rightarrow$ user clicked the item
categorical fields (e.g., gender, location) → one-hot encoding
continuous fields (e.g., age)
$x = [x_{\text{field}_1}, \ldots, x_{\text{field}_j}, \ldots, x_{\text{field}_m}] \rightarrow$ sparse & high-dimensional
$x_{\text{field}_j} \rightarrow$ vector representation of the j-th field of $\mathcal{X}$
$\hat{y} = \text{CTR\_model}(x) \rightarrow$ prediction model
(estimate the probability of a user clicking a specific app in a given context)



"The FM component and deep component share the same feature embedding, which brings two important benefits: 1) it learns both low- and high-order feature interactions from raw features; 2) there is no need for expertise feature engineering of the input, as required in Wide & Deep."



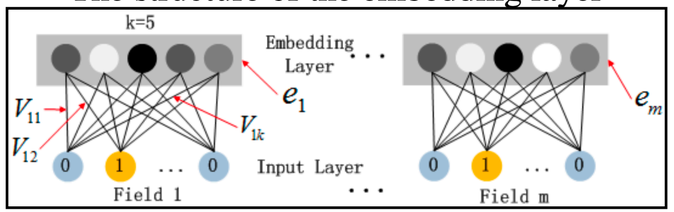


DeepFM
learn both low- and high-order interactions
$\hat{y} = \text{sigmoid}(y_{\text{FM}} + y_{\text{DNN}})$

$$y_{FM} = \langle w, x \rangle + \sum_{j_1=1}^{d} \sum_{j_2=j_1+1}^{d} \langle V_i, V_j \rangle x_{j_1} \cdot x_{j_2}$$

$w \in \mathbb{R}^d, V_i \in \mathbb{R}^k (k \text{ is given})$
$a^{(0)} = [e_1, e_2, \ldots, e_m] \rightarrow$ output of the embedding layer
$m \rightarrow$ number of fields
$e_i \rightarrow$ embedding of the i-th field

The structure of the embedding layer



$a^{(\ell+1)} = \sigma(W^{(\ell)} a^{(\ell)} + b^{(\ell)})$
$y_{\text{DNN}} = \sigma(W^{|H|+1} a^{|H|+1} + b^{|H+1|})$
$|H| \rightarrow$ number of hidden layers

Performance on CTR prediction.

	Company∗		Criteo	
	AUC	LogLoss	AUC	LogLoss
LR	0.8640	0.02648	0.7686	0.47762
FM	0.8678	0.02633	0.7892	0.46077
FNN	0.8683	0.02629	0.7963	0.45738
IPNN	0.8664	0.02637	0.7972	0.45323
OPNN	0.8658	0.02641	0.7982	0.45256
PNN∗	0.8672	0.02636	0.7987	0.45214
LR & DNN	0.8673	0.02634	0.7981	0.46772
FM & DNN	0.8661	0.02640	0.7850	0.45382
DeepFM	**0.8715**	**0.02618**	**0.8007**	**0.45083**

Guo, Huifeng, et al. "DeepFM: a factorization-machine based neural network for CTR prediction." *arXiv preprint arXiv:1703.04247* (2017).

Personalizing Session-based Recommendations with Hierarchical Recurrent Neural Networks

YouTube Video

HRNN: Hierarchical RNN

$S_m = \{i_{m,1}, i_{m,2}, \ldots, i_{m,N_m}\} \rightarrow$ session m

(i: item)

$s_{m,n} = \text{GRU}_{\text{ses}}(i_{m,n}, s_{m,n-1}), n = 1, 2, \ldots, N_m - 1$

(GRU_{ses}: session-level GRU)

$s_{m,n} \rightarrow$ hidden state at step n

$s_{m,0} = 0$

$i_{m,n} \rightarrow$ one-hot vector of the current item ID

$\hat{r}_{m,n} \rightarrow$ output of the RNN
(a score for every item in the catalog)
(the likelihood of being the next item in the session)

$\hat{r}_{m,n} = g(s_{m,n}),\ n = 1, \ldots, N_m - 1$

$g(\cdot)$ is a non-linear function like softmax or tanh

ranking loss functions: cross-entropy, BPR, TOP1

(BPR: Bayesian Personalized Ranking)

TOP1 Loss

$$\frac{1}{N}\sum_{j=1}^{N} I\{\hat{r}_{s,j} > \hat{r}_{s,i}\}$$

($\hat{r}_{s,j}$: score of a sampled irrelevant item; $\hat{r}_{s,i}$: relative rank of the relevant item)

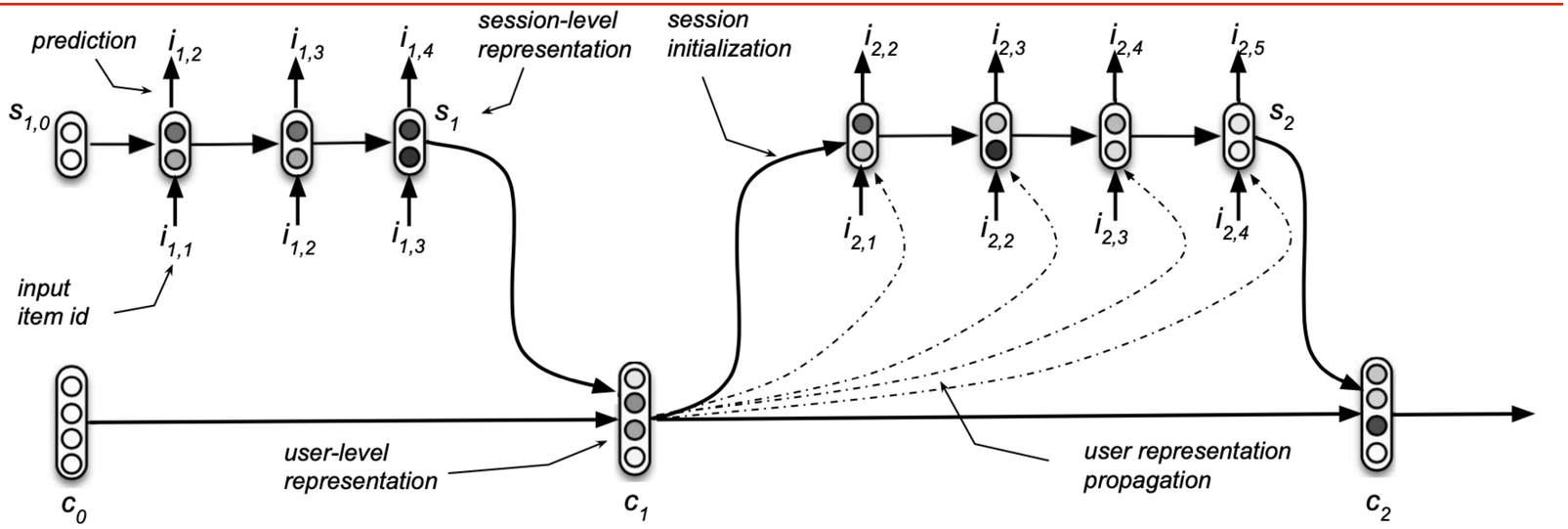
– regularized approximation

$$L_s = \frac{1}{N_S} \cdot \sum_{j=1}^{N_S} \sigma(\hat{r}_{s,j} - \hat{r}_{s,i}) + \sigma(\hat{r}_{s,j}^2)$$

Model user activity across user sessions

$\text{GRU}_{\text{user}} \rightarrow$ user-level GRU

$C^u = \{S_1^u, S_2^u, \ldots, S_{M_u}^u\} \rightarrow$ sessions of user u



$s_1^u, s_2^u, \ldots, s_{M_u}^u \rightarrow$ session-level representation

$s_m^u = s_{m,N_m-1}^u \rightarrow$ last hidden state of GRU_{ses}

$c_m^u \rightarrow$ user-level representation

$c_m = GRU_{usr}(s_m, c_{m-1}),\ m = 1, \ldots, M_u$

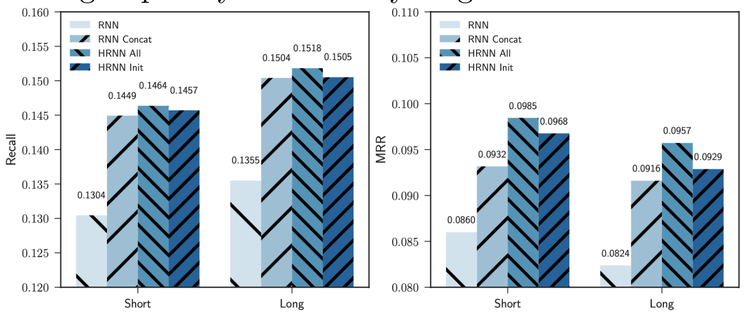
$c_0 = 0$

$s_{m+1,0} = \tanh(W_{init} c_m + b_{init})$

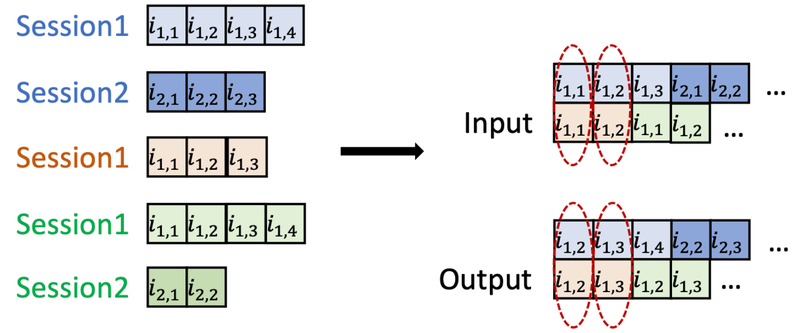
$s_{m+1,n} = GRU_{ses}(i_{m+1,n}, s_{m+1,n-1}\ [, c_m])$ (optional)

$n = 1, \ldots, N_{m+1} - 1$

Recall@5 and MRR@5 on XING dataset grouped by user history length



User-parallel mini-batches



Quadrana, Massimo, et al. "Personalizing session-based recommendations with hierarchical recurrent neural networks." *Proceedings of the Eleventh ACM Conference on Recommender Systems*. 2017.

Variational Autoencoders for Collaborative Filtering

YouTube Video

$u \in \{1, 2, \ldots, U\} \rightarrow$ users

$i \in \{1, 2, \ldots, I\} \rightarrow$ items

$X \in \mathbb{R}^{U \times I} \rightarrow$ user-by-item interaction matrix

$x_u = [x_{u1}, x_{u2}, \ldots, x_{uI}]^T \rightarrow$ click history

$x_{ui} \rightarrow$ number of clicks for item i from user u

For simplicity, let us binarize the click matrix.

$z_u \sim \mathcal{N}(0, I_K)$

$z_u \in \mathbb{R}^K \rightarrow$ latent representation

$x_u \sim \text{Mult}(N_u, \pi(z_u))$

$\pi(z_u) \propto \exp(f_\theta(z_u))$

$\pi(z_u)$ — probability distribution over I items

$N_u = \sum_{i=1}^{I} x_{ui} \rightarrow$ total number of clicks from user u

$\log p_\theta(x_u|z_u) \stackrel{c}{=} \sum_i x_{ui} \log \pi_i(z_u) \rightarrow$ multinomial likelihood

$\log p_\theta(x_u|z_u) \stackrel{c}{=} -\sum_i \frac{c_{ui}}{2}(x_{ui} - f_{ui})^2 \rightarrow$ Gaussian likelihood

$f_\theta(z_u) = [f_{u1}, \ldots, f_{uI}]^T$

$c_{ui} = c_{x_{ui}} \rightarrow$ "confidence weight $(c_1 > c_0)$"

$\log p_\theta(x_u|z_u) \stackrel{c}{=} -\sum_i x_{ui} \log \sigma(f_{ui}) + (1 - x_{ui}) \log(1 - \sigma(f_{ui}))$ — logistic likelihood

$p(z_u|x_u) \rightarrow$ posterior distribution (intractable)

$q(z_u) = \mathcal{N}(\mu_u, \text{diag}(\sigma_u^2)) \rightarrow$ variational distribution

$\text{KL}(q(z_u)\|p(z_u|x_u))$

$g_\phi(x_u) = [\mu_\phi(x_u), \sigma_\phi(x_u)]$

$q_\phi(z_u|x_u) = \mathcal{N}(\mu_\phi(x_u), \text{diag}(\sigma_\phi^2(x_u)))$

$\log p(x_u; \theta) \geq \mathbb{E}_{q_\phi(z_u|x_u)}[\log p_\theta(x_u|z_u)] - \text{KL}[q_\phi(z_u|x_u)\|p(z_u)] := \mathcal{L}(x_u; \theta, \phi)$ — evidence lower bound (ELBO)

$z_u = \mu_\phi(x_u) + \epsilon \odot \sigma_\phi(x_u) \rightarrow$ re-parametrization trick

$\epsilon \sim \mathcal{N}(0, I_K)$

$\mathcal{L}(x_u; \theta, \phi) = \mathbb{E}_{q_\phi(z_u|x_u)}[\log p_\theta(x_u|z_u)] - \beta \text{KL}[q_\phi(z_u|x_u)\|p(z_u)$

$$\theta^{\text{AE}}, \phi^{\text{AE}} = \arg\max_{\theta,\phi} \sum_u \mathbb{E}_{\delta(\mathbf{z_u} - g_\phi(\mathbf{x_u}))}[\log p_\theta(\mathbf{x_u} \mid \mathbf{z_u})]$$

$$= \arg\max_{\theta,\phi} \sum_u \log p_\theta(\mathbf{x_u} \mid g_\phi(\mathbf{x_u}))$$

$\rightarrow$ Regular Autoencoder

– MovieLens-20M (ML-20M)
– Netflix Prize (Netflix)
– Million Song Dataset (MSD)

	ML-20M	Netflix	MSD
# of users	136,677	463,435	571,355
# of items	20,108	17,769	41,140
# of interactions	10.0M	56.9M	33.6M
% of interactions	0.36%	0.69%	0.14%
# of held-out users	10,000	40,000	50,000

	Recall@20	Recall@50	NDCG@100
Mult-VAE$^{\text{PR}}$	**0.395**	**0.537**	**0.426**
Gaussian-VAE$^{\text{PR}}$	0.383	0.523	0.415
Logistic-VAE$^{\text{PR}}$	0.388	0.523	0.419
Mult-DAE	**0.387**	**0.524**	**0.419**
Gaussian-DAE	0.376	0.515	0.409
Logistic-DAE	0.381	0.516	0.414

Partially Regularized VAE with multinomial likelihood

Prediction

$x \rightarrow$ user history

$z = \mu_\phi(x)$ and rank based on $f_\theta(z)$

Metrics

$$\text{Recall@}R(u, \omega) := \frac{\sum_{r=1}^{R} \mathbb{I}[\omega(r) \in I_u]}{\min(R, |I_u|)}$$ → considers all items ranked within the first R to be equally important

$w(r) \rightarrow$ item at rank r

$I_u \rightarrow$ set of held-out items that user u clicked on

$$\text{DCG@}R(u, \omega) := \sum_{r=1}^{R} \frac{2^{\mathbb{I}[\omega(r) \in I_u]} - 1}{\log(r + 1)}$$ → uses a monotonically increasing discount to emphasize the importance of higher ranks versus lower ones

↓ Truncated discounted cumulative gain

NDCG@R is the DCG@R linearly normalized to $[0, 1]$ after dividing by the best possible $DCG@R$, where all the held-out items are ranked at the top.

Liang, Dawen, et al. "Variational autoencoders for collaborative filtering." *Proceedings of the 2018 World Wide Web Conference*. 2018.

Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding

YouTube Video

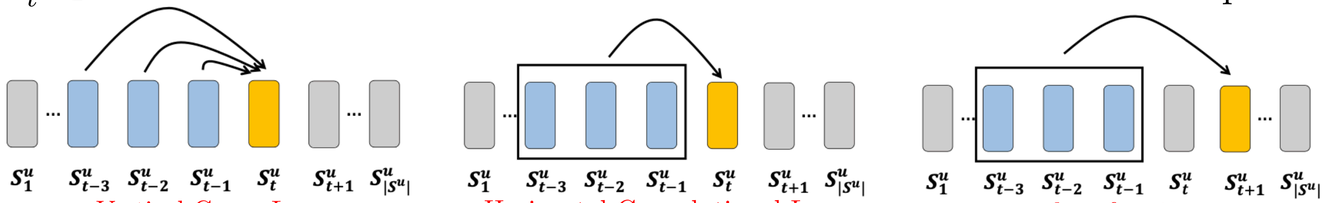
Convolutional Sequence Embedding Recommendation Model (Caser)

$\mathcal{U} = \{u_1, u_2, \ldots, u_{|\mathcal{U}|}\} \rightarrow$ set of users

$\mathcal{I} = \{i_1, i_2, \ldots, i_{|\mathcal{I}|}\} \rightarrow$ universe of items

$\mathcal{S}^u = (\mathcal{S}^u_1, \ldots, \mathcal{S}^u_{|\mathcal{S}^u|}) \rightarrow$ sequence of items from $\mathcal{I}$ associated with user u

$\mathcal{S}^u_t \in \mathcal{I} \rightarrow$ index t denotes the order in which an action occurs in the sequence



(a) point-level **(b) union-level, no skip** **(c) union-level, skip once**

input: L successive items from the user's sequence $\mathcal{S}^u$

targets: next T items

$L + T \rightarrow$ size of the sliding window over the user's sequence

$(u,$ previous L items, next T items$) \rightarrow$ training instance triplet

Embedding Look-up

$Q_i \in \mathbb{R}^d \rightarrow$ embedding for item i $\quad d \rightarrow$ number of latent dimensions

$E^{(u,t)} = [Q_{\mathcal{S}^u_{t-L}}, \ldots, Q_{\mathcal{S}^u_{t-1}}]^T \in \mathbb{R}^{L \times d} \rightarrow$ embedding matrix

$P_u \in \mathbb{R}^d \rightarrow$ user embedding

Regard $E \in \mathbb{R}^{L \times d}$ as an image!

Horizontal Convolutional Layer

$F^k \in \mathbb{R}^{h \times d}, k = 1, \ldots, n \rightarrow n$ horizontal filters

$h \in \{1, 2, \ldots, L\} \rightarrow$ height of the filter

$c^k_i = \phi_c(E_{i:i+h-1} \odot F^k) \rightarrow i$-th convolution value

activation func. $\quad$ inner product

$c^k = [c^k_1, \ldots, c^k_{L-h+1}]$

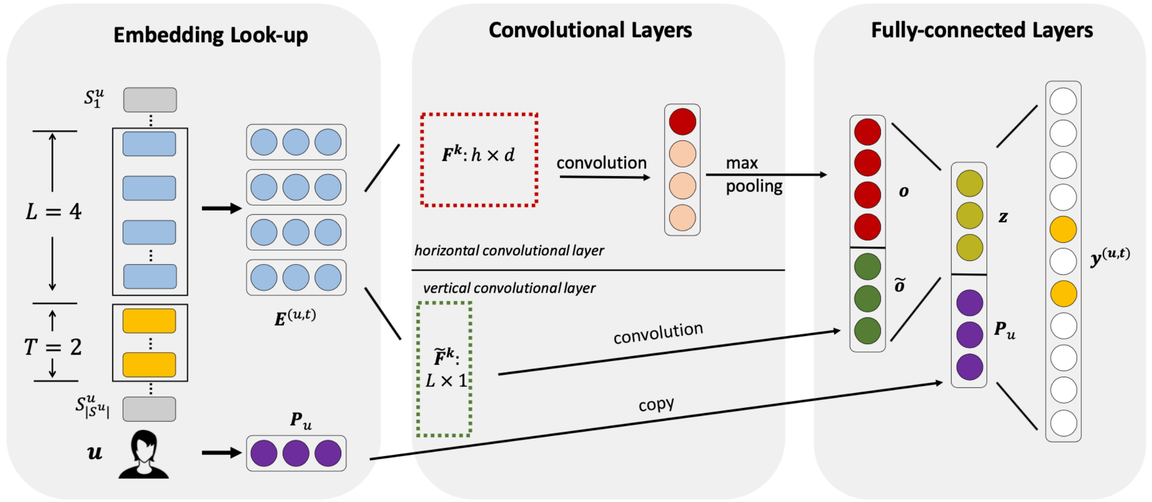
$o = [\max(c^1), \ldots, \max(c^n)]$

Vertical Conv. Layer

$\tilde{F}^k \in \mathbb{R}^{L \times 1}, k = 1, \ldots, \tilde{n}$

$\tilde{c}^k = [\tilde{c}^k_1, \ldots, \tilde{c}^k_d]$

$\tilde{o} = [\tilde{c}^1, \ldots, \tilde{c}^n] \in \mathbb{R}^{d\tilde{n}}$



Network Training

$p(\mathcal{S}^u_t | \mathcal{S}^u_{t-1}, \ldots, \mathcal{S}^u_{t-L}) = \sigma(y^{(u,t)}_{\mathcal{S}^u_t}) \rightarrow$ sigmoid

$\mathcal{C}^u \rightarrow$ collection of time-steps for which we would like to make predictions for user u

$$p(\mathcal{S}|\theta) = \prod_u \prod_{t \in \mathcal{C}^u} \sigma(y^{(u,t)}_{\mathcal{S}^u_t}) \prod_{j \neq \mathcal{S}^u_t} (1 - \sigma(y^{(u,t)}_j))$$ likelihood of all sequences in the dataset

To capture the skip behavior, replace $\mathcal{S}^u_t$ with $\mathcal{D}^u_t$.

$$\ell = \sum_u \sum_{t \in \mathcal{C}^u} \sum_{i \in \mathcal{D}^u_t} -\log(\sigma(\boldsymbol{y}^{(u,t)}_i)) + \sum_{j \neq i} -\log(1 - \sigma(\boldsymbol{y}^{(u,t)}_j)) \rightarrow \text{binary cross-entropy}$$

Evaluation Metrics

$$\text{Prec@}N = \frac{|R \bigcap \hat{R}_{1:N}|}{N} \qquad \text{Recall@}N = \frac{|R \bigcap \hat{R}_{1:N}|}{|R|}$$

average precision

$$\text{AP} = \frac{\sum_{N=1}^{|\hat{R}|} \text{Prec@}N \times \text{rel}(N)}{|\hat{R}|}$$

$\hat{R}_{1:N} \rightarrow$ list of top N predicted items for a user

$R \rightarrow$ last 20% of actions in a user's sequence

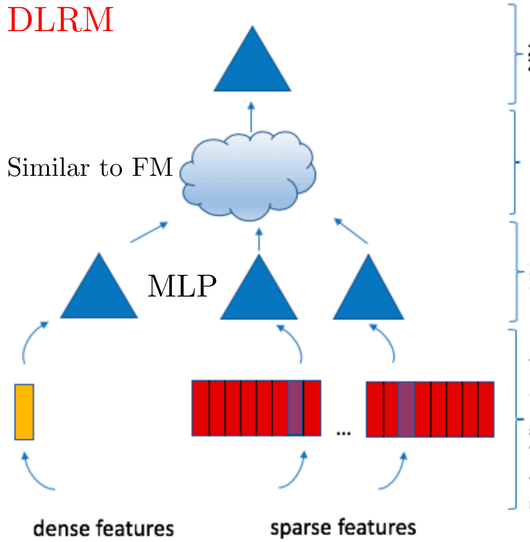
$\text{rel}(N) = 1$ if the N-th item in $\hat{R}$ is in R

Tang, Jiaxi, and Ke Wang. "Personalized top-n sequential recommendation via convolutional sequence embedding." *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 2018.

Deep Learning Recommendation Model for Personalization and Recommendation Systems

YouTube Video

DLRM



Embeddings

$e_i \in \{0,1\}^m \rightarrow$ one-hot vector with the i-th position being 1 while others are 0, where index i corresponds to i-th category

$W \in \mathbb{R}^{m \times d} \rightarrow$ embedding table

$w_i^T = e_i^T W \rightarrow i$-th row of W

$a^T = [0, \ldots, a_{i_1}, \ldots, a_{i_k}, \ldots, 0]$

multi-hot vector (weighted combination of multiple items)

$S = A^T W \rightarrow$ mini-batch of t embedding lookups

$A = [a_1, \ldots, a_t]$

Matrix Factorization (MF)

$\mathcal{S} \rightarrow$ set of users that have rated some products

$w_i \in \mathbb{R}^d, i = 1, \ldots, m \rightarrow$ representation of the i-th product

$v_j \in \mathbb{R}^d, j = 1, \ldots, n \rightarrow$ representation of the j-th user

$\mathcal{S} = \{(i,j) : \text{when the } i\text{-th product has been rated by user } j\}$

$$\min \sum_{(i,j) \in \mathcal{S}} r_{ij} - w_i^T v_j$$

r_{ij}: rating of the i-th product by the j-th user

$R = [r_{ij}]$

$W^T = [w_1, \ldots, w_m] \implies R \approx WV^T$

$V^T = [v_1, \ldots, v_n]$

Factorization Machine (FM)

$\phi : \mathbb{R}^n \rightarrow T$

prediction function

$x \in \mathbb{R}^n \rightarrow$ input data point

$y \in T \rightarrow$ target label

click-through rate (CTR): $T = \{+1, -1\}$ (presence/absence of a click)

$\hat{y} = b + w^T x + x^T \text{upper}(VV^T)x$

$V \in \mathbb{R}^{n \times d}$

strictly upper triangular part

Multilayer Perceptrons (MLP)

$\hat{y} = W_k \sigma(W_{k-1} \sigma(\cdots \sigma(W_1 x + b_1) \cdots) + b_{k-1}) + b_k$

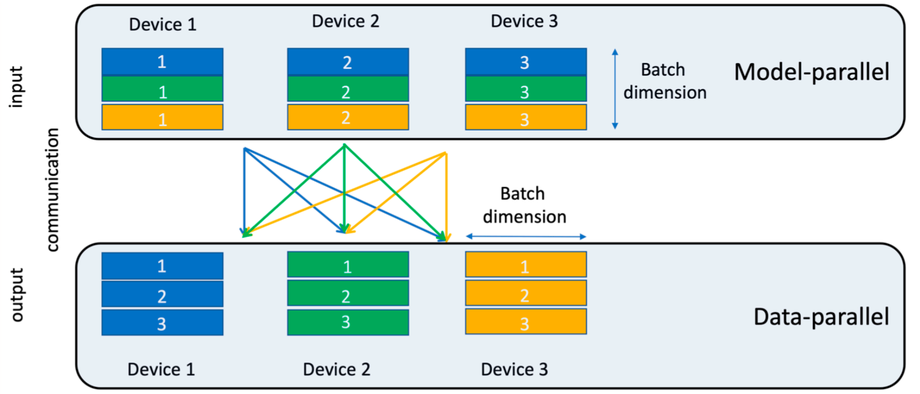
Parallelism

Embeddings $\rightarrow$ Memory Bound (Model Parallel)

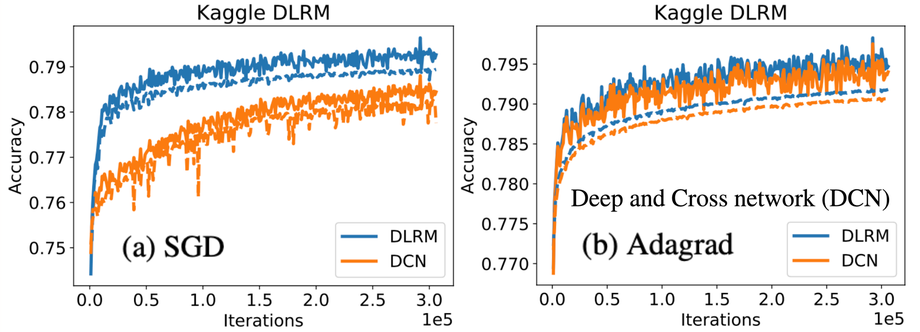
MLP $\rightarrow$ Compute Bound (Data Parallel)

Data

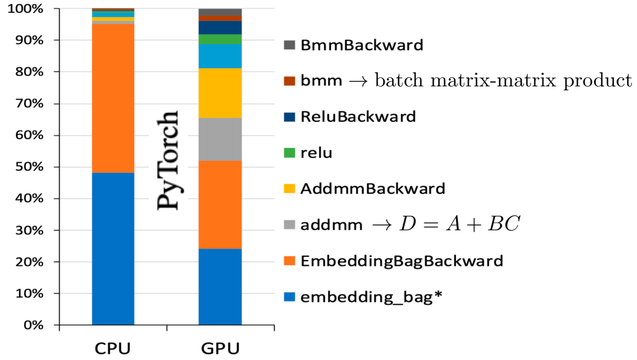
Criteo AI Labs Ad Kaggle and Terabyte data sets for ad CTR prediction



Butterfly shuffle for the all-to-all (personalized) communication



Comparison of training (solid) and validation (dashed) accuracies of DLRM and DCN



Naumov, Maxim, et al. "Deep learning recommendation model for personalization and recommendation systems." *arXiv preprint arXiv:1906.00091* (2019).

Boulder

Questions?

YouTube Playlist